1주차

스프링-입문-스프링부트

섹션 1. 강의 소개

섹션 2. 프로젝트 환경 설정

hello-spring

→ main

→ resources : java를 제외한 모든 파일들

→ test

→ gitignore

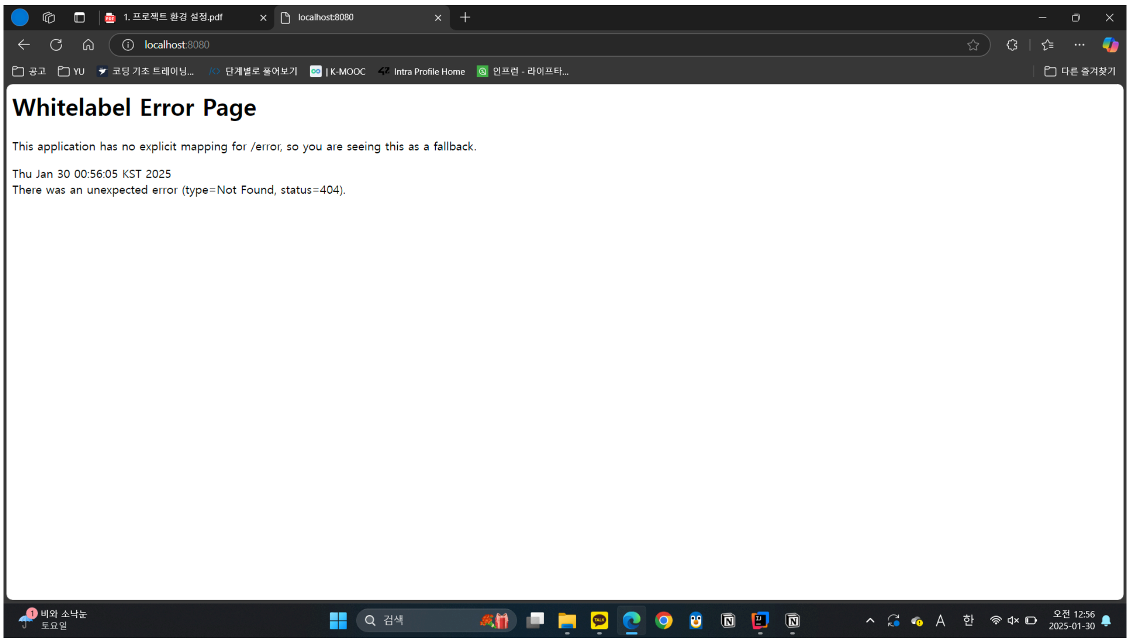
build.gradle

```
plugins { //우리가 선택한 java에 대한 플러그인
    java
    id("org.springframework.boot") version "3.4.2"
    id("io.spring.dependency-management") version "1.1.7"
}

group = "hello"
version = "0.0.1-SNAPSHOT"

java {
    toolchain {
        languageVersion = JavaLanguageVersion.of(21)
    }
}

repositories { //maveCentral에서 라이브러리를 다운받아라
    mavenCentral()
}

dependencies {
    implementation("org.springframework.boot:spring-boot-starter-thymeleaf")
    implementation("org.springframework.boot:spring-boot-starter-web") //처음
    testImplementation("org.springframework.boot:spring-boot-starter-test") //
    testRuntimeOnly("org.junit.platform:junit-platform-launcher")
}

tasks.withType<Test> {
    useJUnitPlatform()
}
```

HelloSpringApplication.java 실행

```
package hello.hello_spring;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class HelloSpringApplication {

    public static void main(String[] args) {
        SpringApplication.run(HelloSpringApplication.class, args);
    }

}
```

SpringBootApplication실행



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Thu Jan 30 00:56:05 KST 2025
There was an unexpected error (type=Not Found, status=404).

라이브러리 살펴보기

핵심 라이브러리

- spring-boot-starter-web
  - tomcat
  - webmvc
- spring-boot-starter-thymeleaf
- spring-boot-starter
  - spring-boot
    - spring-core
  - spring-boot-starter-logging
    - logback, slf4j

테스트 라이브러리

- spring-boot-starter-test
  - junit
  - mockito

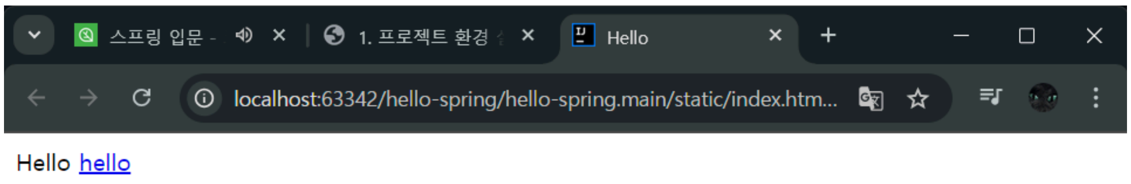
- assertj
- spring-test

view 환경 설정

index.html ⇒ Welcome page

```
<!DOCTYPE HTML>
<html>
<head>
   <title>Hello</title>
   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
Hello
<a href="/hello">hello</a>
</body>
</html>
```

Spring에서는 static/index.html을 올려두면 Welcome page 기능을 제공



localhost:63342/hello-spring/hello-spring.main/static/index.htm...

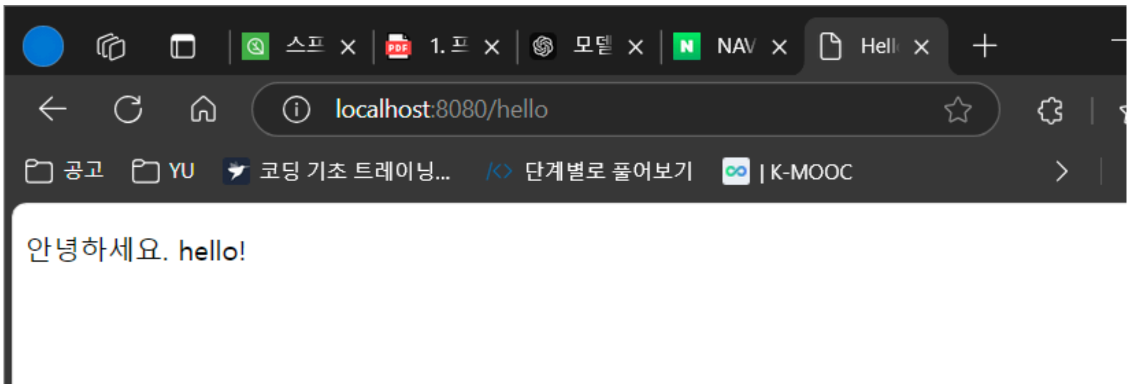
Hello hello

thymeleaf 템플릿 엔진

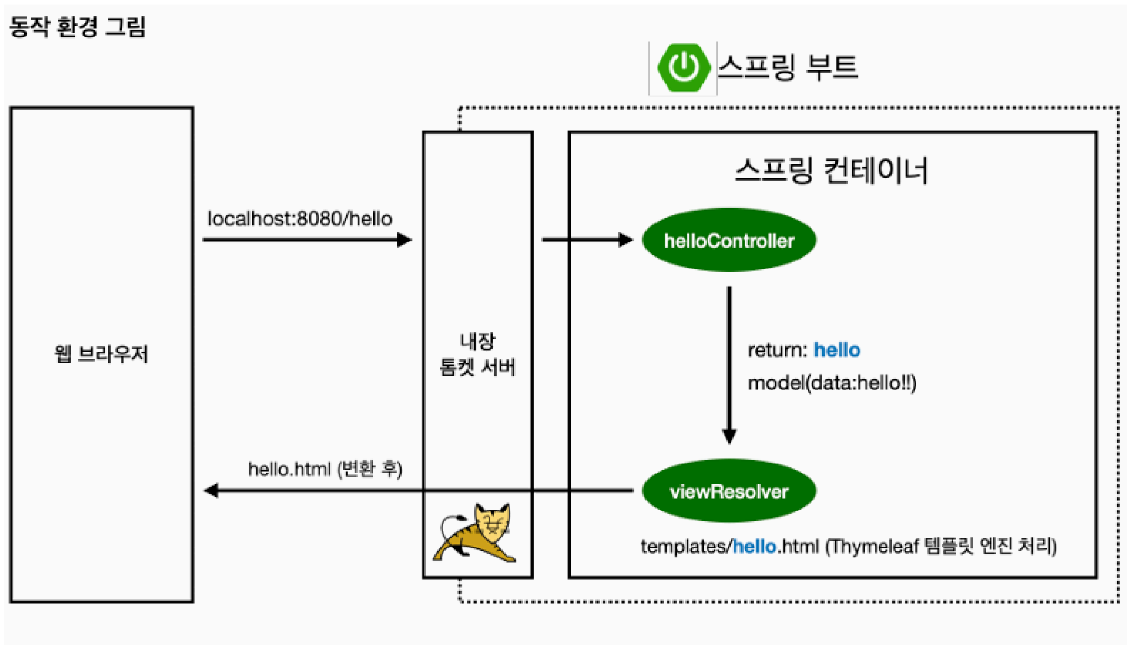
```
@Controller
public class HelloController {
  @GetMapping("hello") / '/hello'로 들어오면 hello메서드 호출
public String hello(Model model) {
     model.addAttribute("data", "hello!!");
return "hello";
```

```
    }
  }
```

```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Hello</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<p th:text="'안녕하세요. ' + ${data}" >안녕하세요. 손님</p> //thymeleaf → data는
</body>
</html>
```



안녕하세요. hello!

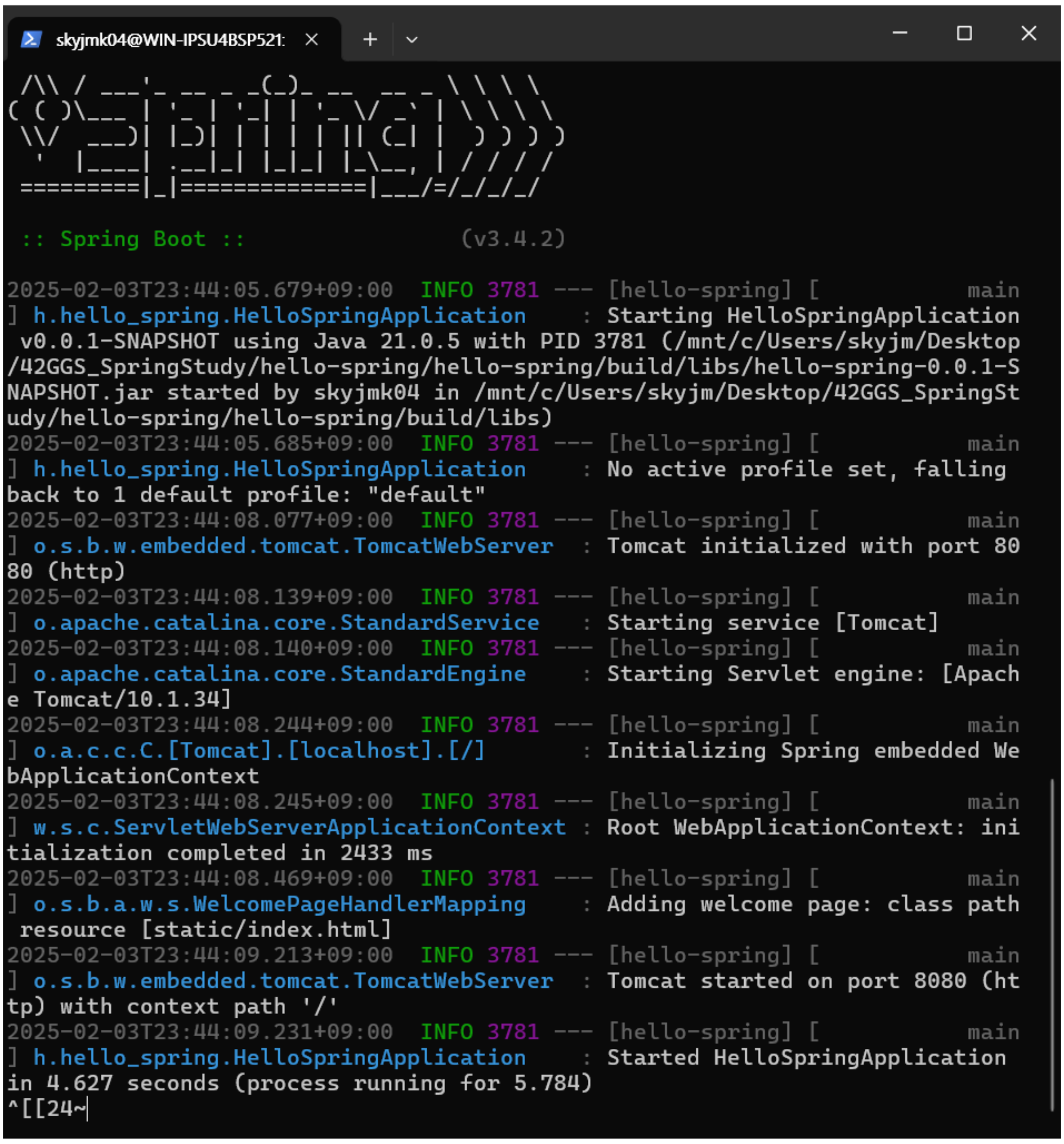


helloController에서 문자를 반환하면 viewResolver가 화면을 찾아서 처리.

빌드하고 실행하기

./gradlew build

java -jar hello-spring-0.0.1-SNAPSHOT.jar



```
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.4.2)

2025-02-03T23:44:05.679+09:00  INFO 3781 --- [hello-spring] [           main
] h.hello_spring.HelloSpringApplication     : Starting HelloSpringApplication
 v0.0.1-SNAPSHOT using Java 21.0.5 with PID 3781 (/mnt/c/Users/skyjm/Desktop
/42GGS_SpringStudy/hello-spring/hello-spring/build/libs/hello-spring-0.0.1-S
NAPSHOT.jar started by skyjmk04 in /mnt/c/Users/skyjm/Desktop/42GGS_SpringSt
udy/hello-spring/hello-spring/build/libs)
2025-02-03T23:44:05.685+09:00  INFO 3781 --- [hello-spring] [           main
] h.hello_spring.HelloSpringApplication     : No active profile set, falling
back to 1 default profile: "default"
2025-02-03T23:44:08.077+09:00  INFO 3781 --- [hello-spring] [           main
] o.s.b.w.embedded.tomcat.TomcatWebServer   : Tomcat initialized with port 80
80 (http)
2025-02-03T23:44:08.139+09:00  INFO 3781 --- [hello-spring] [           main
] o.apache.catalina.core.StandardService    : Starting service [Tomcat]
2025-02-03T23:44:08.140+09:00  INFO 3781 --- [hello-spring] [           main
] o.apache.catalina.core.StandardEngine     : Starting Servlet engine: [Apach
e Tomcat/10.1.34]
2025-02-03T23:44:08.244+09:00  INFO 3781 --- [hello-spring] [           main
] o.a.c.c.C.[Tomcat].[localhost].[/]        : Initializing Spring embedded We
bApplicationContext
2025-02-03T23:44:08.245+09:00  INFO 3781 --- [hello-spring] [           main
] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: ini
tialization completed in 2433 ms
2025-02-03T23:44:08.469+09:00  INFO 3781 --- [hello-spring] [           main
] o.s.b.a.w.s.WelcomePageHandlerMapping    : Adding welcome page: class path
 resource [static/index.html]
2025-02-03T23:44:09.213+09:00  INFO 3781 --- [hello-spring] [           main
] o.s.b.w.embedded.tomcat.TomcatWebServer   : Tomcat started on port 8080 (ht
tp) with context path '/'
2025-02-03T23:44:09.231+09:00  INFO 3781 --- [hello-spring] [           main
] h.hello_spring.HelloSpringApplication     : Started HelloSpringApplication
in 4.627 seconds (process running for 5.784)
^[[24~
```

./gradlew clean build ⇒ 완전히 지우고 다시 build

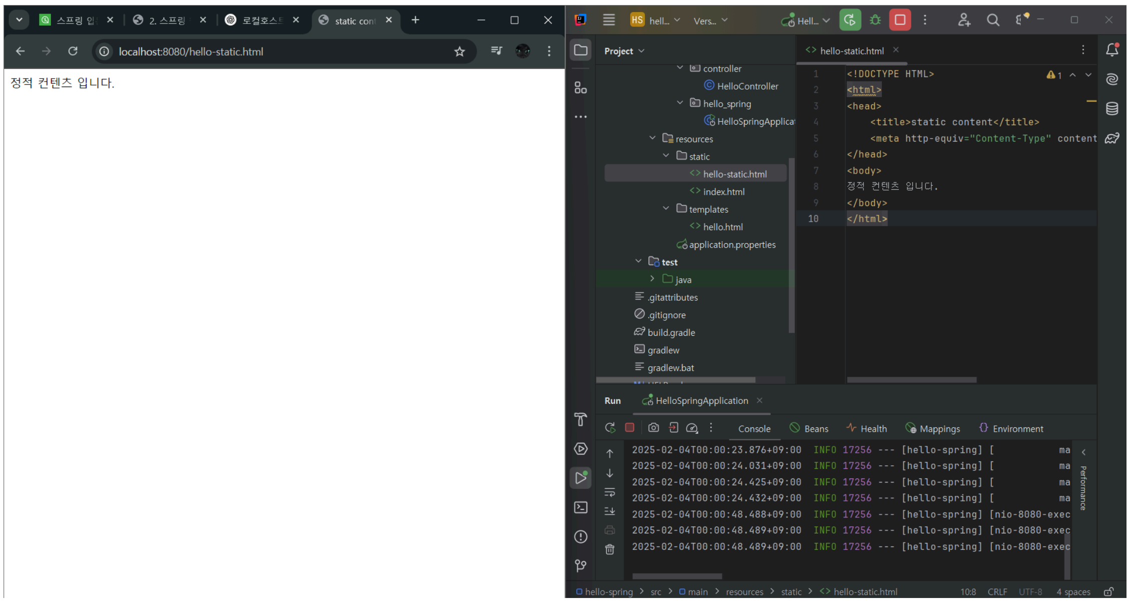
섹션 3. 스프링 웹 개발 기초

정적 컨텐츠

-static 파일이 그대로 반환.

```
<!DOCTYPE HTML>
<html>
<head>
   <title>static content</title>
```

```html
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
</head>
<body>
정적 컨텐츠 입니다.
</body>
</html>
```

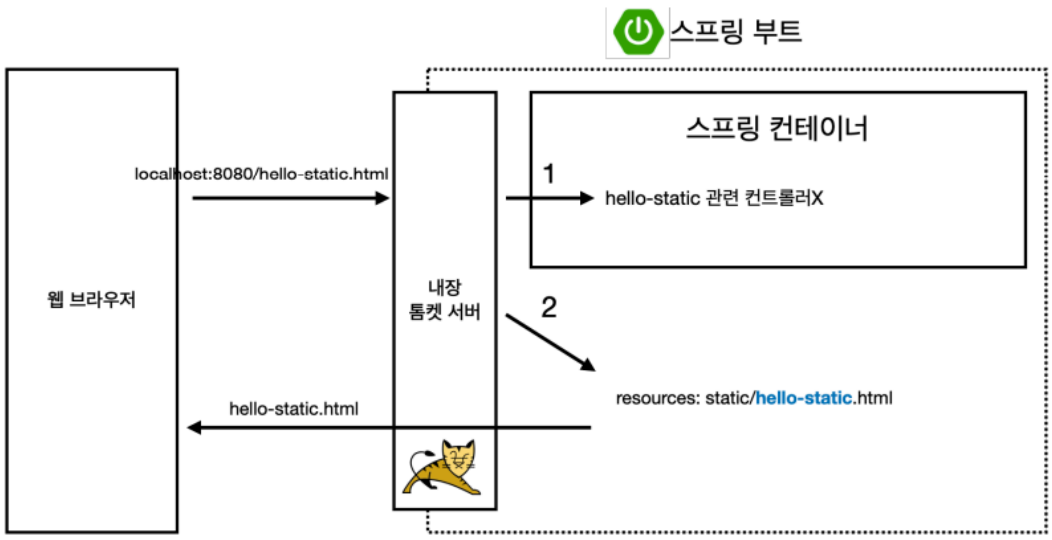


localhost:8080/hello-static.html

정적 컨텐츠 입니다.

```html
<!DOCTYPE HTML>
<html>
<head>
    <title>static content</title>
    <meta http-equiv="Content-Type" content
</head>
<body>
정적 컨텐츠 입니다.
</body>
</html>
```

```
2025-02-04T00:00:23.876+09:00  INFO 17256 --- [hello-spring] [           ma
2025-02-04T00:00:24.031+09:00  INFO 17256 --- [hello-spring] [           ma
2025-02-04T00:00:24.425+09:00  INFO 17256 --- [hello-spring] [           ma
2025-02-04T00:00:24.432+09:00  INFO 17256 --- [hello-spring] [           ma
2025-02-04T00:00:48.488+09:00  INFO 17256 --- [hello-spring] [nio-8080-exec
2025-02-04T00:00:48.489+09:00  INFO 17256 --- [hello-spring] [nio-8080-exec
2025-02-04T00:00:48.489+09:00  INFO 17256 --- [hello-spring] [nio-8080-exec
```


스프링 부트
웹 브라우저
localhost:8080/hello-static.html
내장
톰켓 서버
스프링 컨테이너
1
hello-static 관련 컨트롤러X
2
resources: static/hello-static.html
hello-static.html

컨트롤러가 우선순위 → hello-static.html

MVC와 템플릿 엔진

MVC ⇒ Model, View, Controller

Controller

```
@GetMapping("hello-mvc")
///외부에서 파라미터를 받는다.
    public String helloMvc(@RequestParam("name") String name, Model model
        model.addAttribute("name", name);
        return "hello-template";
    }
```

View

```
<html xmlns:th="http://www.thymeleaf.org">
<body>
<p th:text="'hello ' + ${name}">hello! empty</p>
</body>
</html>
```

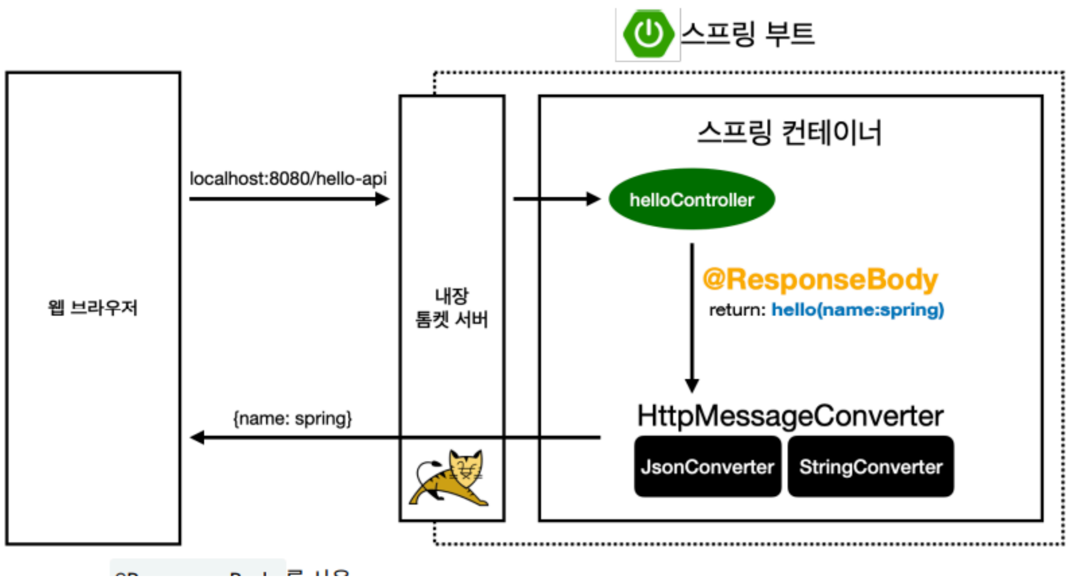
API

```
    @GetMapping("hello-string")
    @ResponseBody //응답 body부분에 내용을 직접 넣겠다.
    public String helloString(@RequestParam(value = "name", required = false)
        return "hello " + name; //view없이 이 문자가 그대로
    }
```

@ResponseBody 객체 반환

```
  @GetMapping("hello-api")
    @ResponseBody
    public Hello helloApi(@RequestParam("name") String name) {
        Hello hello = new Hello();
        hello.setName(name);
        return hello;
    }

    static class Hello{
        private String name;

        public String getName() {
            return name;
        }

        public void setName(String name) {
            this.name = name;
        }
    }
```

기본으로 json으로 반환한다.



스프링 부트
웹 브라우저
localhost:8080/hello-api
내장 톰켓 서버
스프링 컨테이너
helloController
@ResponseBody
return: hello(name:spring)
HttpMessageConverter
JsonConverter
StringConverter
{name: spring}